Programming Languages and Data Access Methods

August 2023

Manufacturer	Programming Language Name	Version	Current Classification	General Availability	Projected	Approved	Divest: Plan	Divest: Execution	Prohibited	Vendor End of Support	References	Notes
General recommendation											https://media.defense.gov/2022/Nov/10/2003112742/-1/-1/0/CSI_SOFTWARE_MEMORY_SAFETY.PDF https://insights.sei.cmu.edu/blog/rust-software-security-a-current-state-assessment/ https://survey.stackoverflow.co/2022/	**EA Recommendation:** Use of a "memory safe" language is recommended when possible. Examples of memory safe language include: C#, Go, Java®, Ruby™, Rust®, and Swift®. Microsoft® revealed at a conference in 2019 that from 2006 to 2018, 70 percent of their vulnerabilities were due to memory safety issues. Google® also found a similar percentage of memory safety vulnerabilities over several years in Chrome®. Malicious cyber actors can exploit these vulnerabilities for remote code execution or other adverse effects, which can often compromise a device and be the first step in large-scale network intrusions. Commonly used languages, such as C and C++, provide a lot of freedom and flexibility in memory management while relying heavily on the programmer to perform the needed checks on memory references. Simple mistakes can lead to exploitable memory-based vulnerabilities. Software analysis tools can detect many instances of memory management issues and operating environment options can also provide some protection, but inherent protections offered by memory safe software languages can prevent or mitigate most memory management issues. NSA recommends using a memory safe language when possible. While the use of added protections to non-memory safe languages and the use of memory safe languages do not provide absolute protection against exploitable memory issues, they do provide considerable protection. Therefore, the overarching software community across the private sector, academia, and the U.S. Government have begun initiatives to drive the culture of software development towards utilizing memory safe languages. https://endoflife.date/
		NA	NA	NA	NA	NA	NA	NA	NA	NA		
Oracle	PL/SQL	All versions	Approved	1995							https://en.wikipedia.org/wiki/PL/SQL	Oracle: https://www.oracletutorial.com/plsql-tutorial/what-is-plsql/ Tech on the Net: https://www.techonthenet.com/oracle/index.php https://endoflife.date/
Oracle	Java	26 (LTS)	**Projected**		2027						End of Life Info: https://endoflife.date/java	https://en.wikipedia.org/wiki/Java_(programming_language) https://endoflife.date/
		20 (LTS)	**Projected**		2024						End of Life Info: https://endoflife.date/java	https://en.wikipedia.org/wiki/Java_(programming_language)
		21	**Skipped**		9/22/2023							Note: Tracking N and N-1 to LTS versions only while skipping all other version between LTS versions.
		20	**Skipped**		3/22/2023							
		19	**Skipped**		9/22/2022							
		18	**Skipped**		3/22/2022							
		17 (LTS)	Approved	9/14/2021	N	12/31/2021	9/30/2025	9/30/2026	9/30/2027	9/30/2029	End of Life Info: https://endoflife.date/java	Java, as developed by the OpenJDK Project, which is owned and primarily employed by Oracle, has been on a 6-month rapid-release cycle since the release of Java 10. Starting with Java 11, new LTS releases occur every six releases, or three years. Java 8 is the last release on the old cycle methodology still in active support. Non-LTS releases are supported for 6 months. The latest supported release in each release cycle can be found at https://www.oracle.com/java/technologies/java-se-glance.html.
		16	**Skipped**	3/16/2021					**Prohibited 9/30/2021**	9/30/2021		Note: Tracking N and N-1 to LTS versions only while skipping all other version between LTS versions.
		15	**Skipped**	9/16/2020					**Prohibited 3/31/2021**	3/31/2021		
		14	**Skipped**	3/17/2020					**Prohibited 9/30/2021**	9/30/2020		
		13	**Skipped**	9/17/2019					**Prohibited 3/31/2021**	3/31/2021		
		12	**Skipped**	3/19/2019					**Prohibited 9/30/2019**	9/30/2019		
		11 (LTS)	Approved	9/25/2018	N-1	3/31/2019	9/30/2023	9/30/2024	9/30/2025	9/30/2026	End of Life Info: https://endoflife.date/java	Java, as developed by the OpenJDK Project, which is owned and primarily employed by Oracle, has been on a 6-month rapid-release cycle since the release of Java 10. Starting with Java 11, new LTS releases occur every six releases, or three years. Java 8 is the last release on the old cycle methodology still in active support. Non-LTS releases are supported for 6 months. The latest supported release in each release cycle can be found at https://www.oracle.com/java/technologies/java-se-glance.html.
		10	**Skipped**	3/20/2018					**Prohibited 9/25/2018**	9/25/2018		Note: Tracking N and N-1 to LTS versions only while skipping all other version between LTS versions.
		9	**Skipped**	9/21/2017					**Prohibited 3/20/2018**	3/20/2018		
		8 (LTS)	**Prohibited**	3/18/2014					**Prohibited 3/31/2022**	3/31/2025	End of Life Info: https://endoflife.date/java	Java, as developed by the OpenJDK Project, which is owned and primarily employed by Oracle, has been on a 6-month rapid-release cycle since the release of Java 10. Starting with Java 11, new LTS releases occur every six releases, or three years. Java 8 is the last release on the old cycle methodology still in active support. Non-LTS releases are supported for 6 months. The latest supported release in each release cycle can be found at https://www.oracle.com/java/technologies/java-se-glance.html.
		7 and earlier	**Prohibited**	7/7/2011					**Prohibited Jul-31-2019**	7/31/2019		First version GA May 1995 from Sun Microsystems. https://en.wikipedia.org/wiki/Java_(programming_language)

Programming Languages and Data Access Methods

Manufacturer	Programming Language Name	Version	Current Classification	General Availability	Projected	Approved	Divest: Plan	Divest: Execution	Prohibited	Vendor End of Support	References	Notes
				August 2023								
CII Honewell Bull	Ada	2012	Prohibited	2012							https://en.wikipedia.org/wiki/Ada_(programming_language)	https://en.wikipedia.org/wiki/Groupe_Bull https://curlie.org/Computers/Programming/Languages/Ada
Association for Computing Machinery (ACM)	ALGOL	DG/L	Prohibited	1972							https://en.wikipedia.org/wiki/ALGOL	
IBM	APL	APL2	Prohibited	1984							https://en.wikipedia.org/wiki/APL_(programming_language)	
IBM	Assembler	BAL	Prohibited	1964							https://en.wikipedia.org/wiki/IBM_Basic_Assembly_Language_and_successors	
Dartmouth	BASIC	All versions	Prohibited	1964							https://en.wikipedia.org/wiki/BASIC	
Bell Labs	"C"	All versions	Prohibited	1973							https://en.wikipedia.org/wiki/The_C_Programming_Language	https://www.section.io/engineering-education/history-of-c-programming-language/
Nantucket Corporation	Clipper	xBase	Prohibited	1997							https://en.wikipedia.org/wiki/Clipper_(programming_language)	aka CA-Clipper
CODASYL	COBOL	All versions	Prohibited	1959							https://en.wikipedia.org/wiki/COBOL	https://en.wikipedia.org/wiki/CODASYL
Embarcadero Technologies	Delphi	All versions	Prohibited	1995							https://en.wikipedia.org/wiki/Delphi_(software)	https://en.wikipedia.org/wiki/History_of_Delphi_(software)
IBM	Fortran	All versions	Prohibited	1957							https://en.wikipedia.org/wiki/Fortran	
Software AG	Natural		Prohibited	1979							https://en.wikipedia.org/wiki/ADABAS#Natural_(4GL)	https://en.wikipedia.org/wiki/Natural_language https://www.xenonstack.com/blog/evolution-of-nlp/
MIT	Lisp	All LISP-based languages	Prohibited	1958							https://en.wikipedia.org/wiki/Lisp_(programming_language)	https://en.wikipedia.org/wiki/List_of_programming_languages_by_type#List-based_languages_%E2%80%93_LISPs
Unisys / Sperry	Mapper	All versions	Prohibited	1975							https://en.wikipedia.org/wiki/MAPPER	
Corel	Paradox	All versions	Prohibited	1985							https://en.wikipedia.org/wiki/Paradox_(database)	
Berkley	Pascal	All versions	Prohibited	1970							https://en.wikipedia.org/wiki/Pascal_(programming_language)	https://www.britannica.com/technology/Pascal-computer-language
IBM	PL/I	All versions	Prohibited	1964							https://en.wikipedia.org/wiki/PL/I	aka PL/1
IBM	Rexx	All versions	Prohibited	1979							https://en.wikipedia.org/wiki/Rexx	
IBM	RPG X	All versions	Prohibited	1959							https://en.wikipedia.org/wiki/IBM_RPG	

Programming Languages and Data Access Methods

August 2023

Manufacturer	Programming Language Name	Version	Current Classification	General Availability	Projected	Approved	Divest: Plan	Divest: Execution	Prohibited	Vendor End of Support	References	Notes

The following **data access methods** are considered *Prohibited* technologies by COV per Enterprise Technical Architecture (ETA):

https://www.vita.virginia.gov/media/vitavirginiagov/it-governance/ea/pdf/Legacy-IT-Solutions-Topic-Report.pdf

Vendor	Data Access Name	Version	Current Classification	General Availability	Divest: Plan	Divest: Execution	Divest: Plan	Divest: Execution	Prohibited	Vendor End of Support	References	Notes
	Adabas		Prohibited								https://documentation.softwareag.com/natural/nat6313win/pg/pg_dbms_ada.htm	https://en.wikipedia.org/wiki/Access_method
	IMS		Prohibited								https://flylib.com/books/en/2.869.1.57/1/	https://en.wikipedia.org/wiki/Access_method https://flylib.com/books/en/2.869.1.58/1/ https://en.wikipedia.org/wiki/IBM_Information_Management_System
	VSAM		Prohibited								https://en.wikipedia.org/wiki/Virtual_Storage_Access_Method	https://en.wikipedia.org/wiki/Access_method
	ISAM		Prohibited								https://en.wikipedia.org/wiki/ISAM	https://en.wikipedia.org/wiki/Access_method
	xBase		Prohibited								https://en.wikipedia.org/wiki/Microsoft_Data_Access_Components	https://en.wikipedia.org/wiki/Access_method
	Paradox		Prohibited								https://en.wikipedia.org/wiki/Microsoft_Data_Access_Components	https://en.wikipedia.org/wiki/Access_method
	Hierarchical Database Access	All	Prohibited								https://en.wikipedia.org/wiki/Access_method	e.g. HSAM, HISAM, HIDAM, HDAM, PHDAM, SHSAM, SHISAM, etc.
	Network Database Access	All	Prohibited								https://en.wikipedia.org/wiki/Access_method	e.g. BTAM, QTAM, TCAM, VTAM, etc.
	Non-security patch updated data a	All	Prohibited								https://en.wikipedia.org/wiki/Access_method	All data access methods and versions without security patching

The following represents COV **programming language classifications** per Enterprise Technical Architecture (ETA):

https://www.vita.virginia.gov/media/vitavirginiagov/it-governance/ea/pdf/Legacy-IT-Solutions-Topic-Report.pdf

Key	Description
	Emerging
	Projected
	Approved
	Divest: Plan
	Divest: Execute
	Prohibited
	Skipped